



**R2E
of
America**

**MICRAL C MICROCOMPUTER SYSTEM
OPERATOR'S MANUAL**

**TM-1001
Issued October 1977**

**3406 University Avenue S.E.
Minneapolis, Minnesota 55414
(612) 378-7060**

MICRAL C MICROCOMPUTER SYSTEM
OPERATOR'S MANUAL

TM-1001
Issued October 1977

R2E OF AMERICA
3406 University Ave., S.E.
Minneapolis, Minnesota 55414

RECORD OF REVISIONS

Original Issue -- October 1977

TABLE OF CONTENTS

CHAPTER 1. SYSTEM DESCRIPTION

	<u>Page</u>
1.1 General Description	1-1
1.2 Minifloppy Disc Drives	1-2
1.2.1 Minidiskettes	1-2
1.2.2 Proper Insertion of a Minidiskette	1-3
1.3 The MICRAL C Keyboard	1-3
1.4 System Control Switches and Indicators	1-5
1.5 Power Switches and Breaker	1-6

CHAPTER 2. OPERATING INSTRUCTIONS -- BAL

2.1 Introduction	2-1
2.2 Types of Diskettes	2-1
2.3 Bringing Up The System	2-1
2.4 Powering The System Down	2-3
2.5 Copying And Formatting	2-3
2.5.1 Formatting	2-3
2.5.2 Copying A Diskette	2-4
2.6 Creating A Source Program	2-5
2.7 Translating A Program	2-6
2.8 Program Execution	2-9
2.9 Editing A Program	2-10
2.10 The Debug Package	2-13
2.11 The Utility Routines	2-14
2.11.1 Routines For Handling Source Files	2-14
2.11.2 Routines For Handling Various Files	2-15

CHAPTER 3. OPERATING INSTRUCTIONS -- FORTRAN

3.1 Introduction	3-1
3.2 Loading The System	3-1
3.3 Editing A FORTRAN Program	3-2
3.4 Program Compilation	3-3
3.5 Program Execution	3-4

APPENDIX A. MICRAL BAL ERROR LIST

A.1 General	A-1
A.2 Errors Found During Execution of a Program	A-1
A.3 Errors Found During Translation of a Program	A-2
A.4 Errors Found When Using Peripherals	A-2

APPENDIX B. MICRAL FILE SYSTEM RESPONSES

B-1

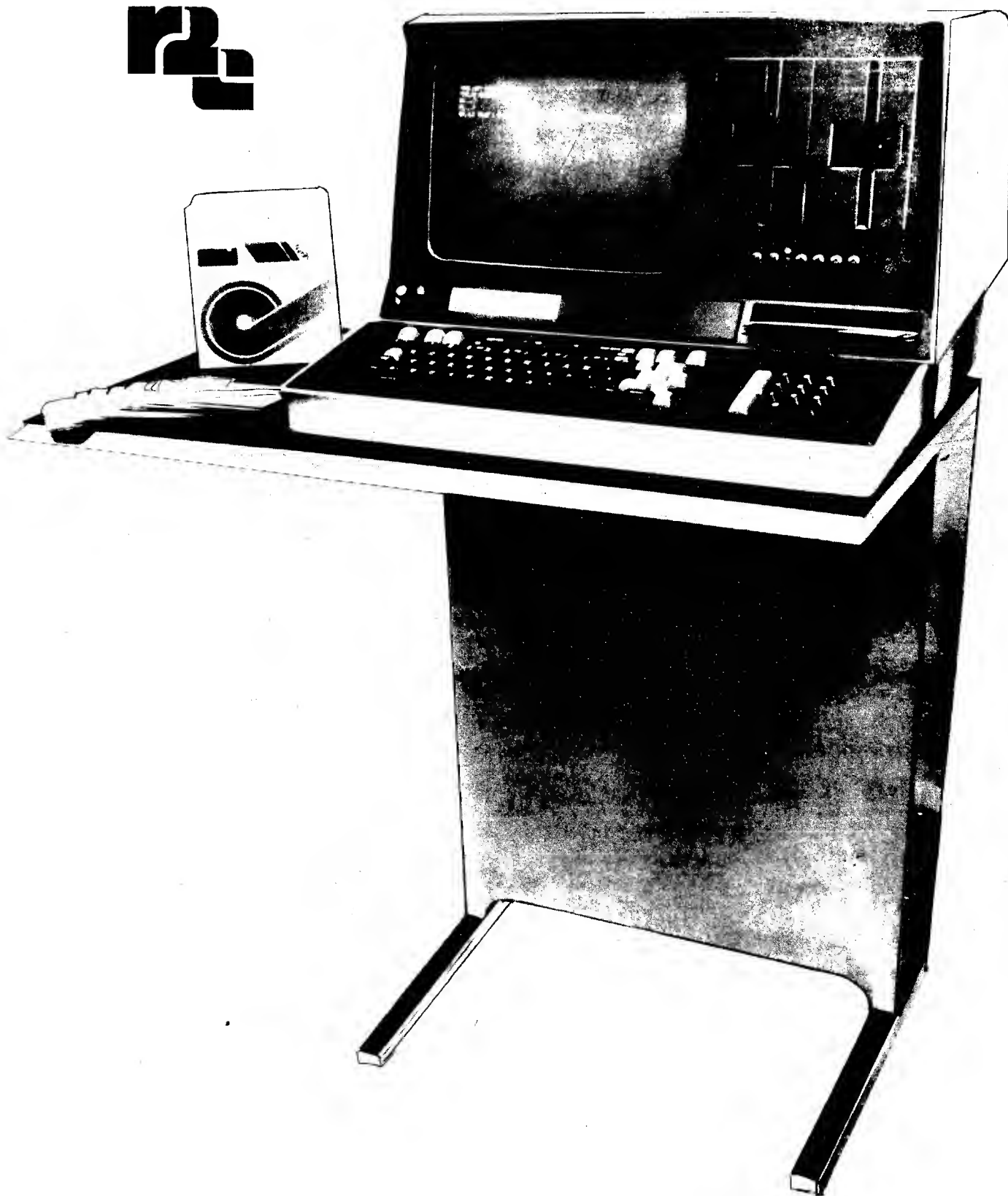


Figure I — MICRAL C System

PREFACE

PURPOSE

This manual provides all instructions and data necessary to operate the R2E MICRAL C Small Business Microcomputer System using either the BAL or FORTRAN Languages.

CONVENTIONS USED IN THIS DOCUMENT

Most (but not all) of the operator commands for the system must be terminated by a carriage return. Whenever required, this is indicated in the text by the symbol (cr).

REFERENCES

For a detailed description of the BAL Language (Business Application Language - A subset of BASIC), refer to the MICRAL C BAL Reference Manual. This document explains all the programming conventions and formats and provides a detailed description of the command structure of the language, plus examples.

This manual also contains a description of the file management system, describes the command structure, and includes examples.

A detailed description of FORT//80 (8080 FORTRAN IV) can be found in the FORT//80, FORTRAN IV Reference Manual.

REVISIONS TO THIS DOCUMENT

The descriptions and specifications herein are subject to change without notice. A comment sheet/mailer is bound into the back of this document. Use it to report comments/corrections to the manual, and/or to place your name on the mailing list for any future updates to this manual.

CHAPTER 1. SYSTEM DESCRIPTION

1.1 GENERAL DESCRIPTION

The MICRAL C is a compact distributed data processing system for business and industry.

It is a free standing system, at home with any office decor. It requires no special power wiring or air conditioning.

The system is composed of an 8080 CPU, 24K of RAM (up to 64K total in 8K increments), two Shugart Minifloppy* drives providing 140K of usable on-line storage, a 1920 character upper/lower case CPU, an ASCII keyboard with separate 13 key numeric pad, a parallel printer interface, a power supply and cabinet.

The printer interface is compatible with various printers supplied by the user, including 30 to 60 cps Xerox-Diablo printers and 180 cps to 1000 lpm Centronics printers.

Figure 1 illustrates the system. It stands 1092 mm (43 in) high, is 851 mm (33½ in) wide and 508 mm (20 in) deep. It weighs about 80 kg (176 lbs).

The system (not including printer) requires 110 Vac, 60 Hz power, at 5 amps, maximum.

*Registered trademark of Shugart Associates

1.2 MINIFLOPPY DISK DRIVES

The system uses two Shugart SA400 minifloppy drives, mounted side by side on the right of the CRT Display. These are rugged mechanical drives using stepping motor actuators. The drives include a write protect feature.

In all operating instructions, the left hand drive will be referred to as Unit 0; the right hand drive is Unit 1.

1.2.1 Minidiskettes

The minifloppy diskette, illustrated in figure 1-2, stores up to 70K of programs or data as formatted by R2E, using Shugart SA105 hard-sectored minidiskettes*, or equivalent. Each 130 mm (5½ in) square minidiskette will store up to 110K bytes, unformatted.

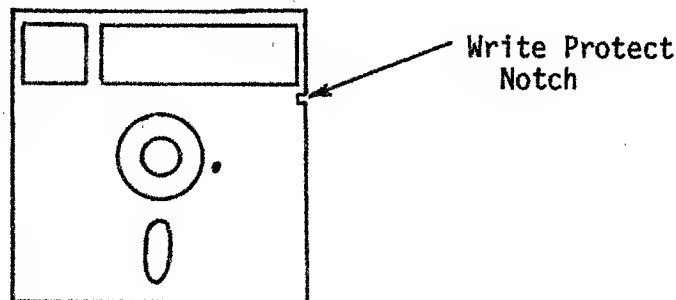


Figure 1-2. Minidiskette

Each minidiskette has 35 tracks, with 16 sectors of 128 characters per track. The SA105 is a hard-sectored diskette--16 sectors (sector holes) per diskette.

NOTE: Do not attempt to use 10 sector hard-sectored diskettes in this system. They will not operate properly with the system software.

The SA105 minidiskette has a write protect capability. A Write Protect Notch is located on the diskette jacket as shown in figure 1-2. When the notch is open, writing is allowed. When the notch is covered with a Write Protect Tab, writing is inhibited.

1.2.2 Proper Insertion of a Minidiskette

Figure 1-3 illustrates the proper insertion of a minidiskette into a drive. Note the orientation of the slot for reading and the orientation of the write protect notch. The label of the inserted diskette must face toward the door opening mechanism.

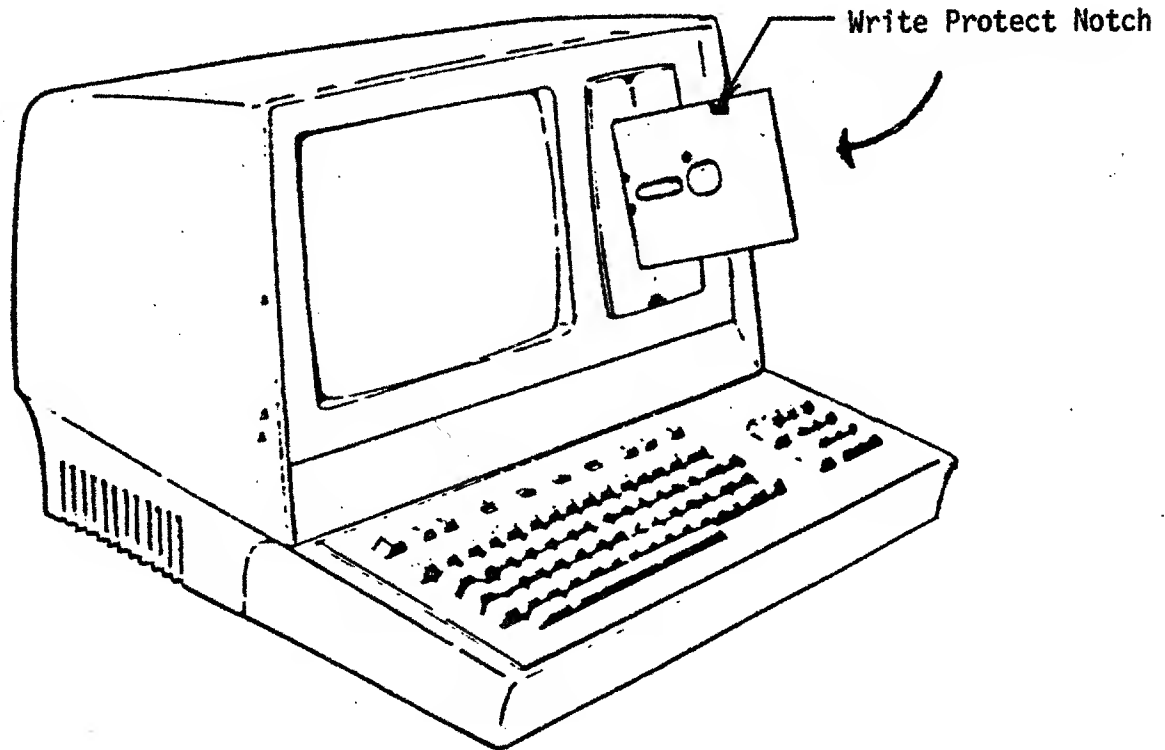


Figure 1-3. Inserting A Minidiskette

1.3 THE MICRAL C KEYBOARD

Figure 1-4 illustrates the MICRAL C keyboard. This is a standard upper/lower case ASCII keyboard with a 13 key numeric pad for convenient entry of numbers. When in Upper case only, the keyboard uses the standard 64 character ASCII character set. When in upper/lower case, it uses a 96 character set.


Most of the keys are self explanatory. The functions of a few special keys and the control keys are explained on the next page.



Figure 1-4. The MICRAL C Keyboard

CONTROL KEY	FUNCTION
CLEAR	Clears the display screen and returns the cursor to the upper left corner position.
BREAK	Unused.
C O	A small LED in this key lights when the keyboard is selected for 96 character ASCII.
PARITY ODD, EVEN, NO	Operate the MICRAL C with this switch in the NO position.
FULL/HALF DUP	Operate with switch in the FULL DUP (Duplex) position.
ON LINE/LOCAL	ON-LINE places the keyboard on-line in communication with the CPU. LOCAL allows data typed on the keyboard to be displayed on the CRT, but not communicated with the CPU.
HIGH RATE/300 /LOW RATE	Set this switch at HIGH RATE for normal system display.
96/64	Selects the character set to be used. Upper/lower case is the 96 character set; upper case only is the 64 character set.

<u>CONTROL KEY</u>	<u>FUNCTION</u>
PAGE	With this locking switch up (normal position) the screen is in scrolling mode, i.e., new lines are added at the bottom of the screen with the top line scrolled off. With this switch locked down, when the screen is full data entry will begin again from the top of the screen. The data previously displayed will not be cleared, however.

<u>KEY</u>	<u>FUNCTION</u>
ESC	Escape -- Used as the "Interruption" key, to interrupt the current operation and return to the beginning of the routine or BAL.
CARRIAGE RETURN	Used to terminate most keyboard entries. Provides both a carriage return (cursor moved to beginning of current line) and a line feed (cursor moved down one line) when pressed.
LINE FEED	Moves cursor down one line, but does not return it to the beginning of the line.
	Moves cursor one character position in the direction indicated. These switches are useful only in system dialog, not in response to an ASK instruction.

1.4 SYSTEM CONTROL SWITCHES AND INDICATORS

There is a panel of system control switches and indicators located immediately below the minifloppy drives. It is illustrated in figure 1-5.

<u>SWITCH OR INDICATORS</u>	<u>FUNCTION</u>
IT	Unused.
DEPART/INIT	<p>DEPART- When raised to the DEPART position, the system loads a program and begins executing it. The program loaded is determined by the four sense or TEST switches. Normally all four switches will be down and the system will load BAL from track 0 of Unit 0.</p> <p>INIT- When pressed to the INIT position, the system returns to BAL.</p>

SWITCH OR INDICATORS	FUNCTION
INDICATORS	
E, P, A	E - Execution - indicates that the processor is running.
	P - Pause - indicates that the processor is waiting for memory or I/O.
	A - Arret (stop) - indicates that the processor is stopped.
SWITCHES	
4, 3, 2, 1	Switch 1 - Selects the drive from which the system will load. Down = Unit 0; Up = Unit 1.

Switches 2,3,4 - Select the track from which the binary program is read, as follows:

SW4	SW3	SW2	TRACK
Down	Down	Down	0
Down	Down	Up	1
Down	Up	Down	2
Down	Up	Up	3
Up	Down	Down	4
Up	Down	Up	5
Up	Up	Down	6
Up	Up	Up	7

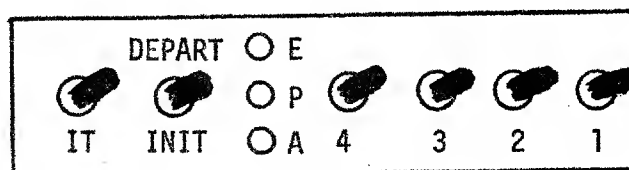


Figure 1-5. System Control & Sense Switches

1.5 POWER SWITCHES AND BREAKER

The system power switches are located on the left hand side of the pedestal under the work surface. In addition, a circuit breaker is located on the lower left rear of the CRT Display. This breaker controls the CRT only.

SWITCH	FUNCTION
GREEN POWER SWITCH	When pressed and locked in position, this switches system power on.
RED POWER OFF SWITCH	When pressed, this releases the POWER switch and removes power from the system.
CRT CIRCUIT BREAKER	In the ON (UP position) powers the CRT Display. In the OFF (DOWN position) removes CRT power.

CHAPTER 2. OPERATING INSTRUCTIONS -- BAL

2.1 INTRODUCTION

This chapter contains complete descriptions of operating instructions for BAL, the Editor, Utilities, and File Management system. Instructions for operation of the FORTRAN Compiler are found in Chapter 3.

2.2 TYPES OF DISKETTES

In the course of using the MICRAL C for various programs, you will use and/or generate the following types of diskettes containing programs and data:

1. System Diskette -- Two BAL system diskettes are provided with your MICRAL C. They each contain a complete copy of the operating system. We suggest that you write-protect these diskettes and use them only for making copies of the operating system.
2. Execution Diskette -- This diskette is produced during translation of your program. It contains the BAL run-time package and your executable translated program.
3. Program Source Diskette -- Contains your source program, written in BAL. It is this source program which is translated to produce an executable program.
4. File Diskettes -- These diskettes contain the files (as necessary) used by your user program(s). Each diskette may contain several files of the same or different types.

These different types of diskettes are discussed in more detail in following paragraphs.

2.3 BRINGING UP THE SYSTEM

Follow the steps below to power the system up and bring the BAL system on-line.

1. Press the Green POWER button until it locks into place. The button is mounted on the upper left corner of the pedestal, just under the work surface.
2. The red and green lights (E & P) on the system control panel will light, and the CRT will come on, provided that the circuit breaker (located at the left rear) is ON. In a second or so, the CRT will warm up and you will see the blinking cursor.

<u>BAL Command</u>	<u>Function</u>	<u>Section</u>
D(cr)	DEBUG -- Loads the Debug package which is then used to trace program execution and debug the BASIC program.	2.10
C(cr)	COMMAND -- Calls various utility programs such as Dump, Formatter, etc.	2.11

2.4 POWERING THE SYSTEM DOWN

To power down, remove the diskettes from the minifloppy drives and press the Red button located immediately below the POWER button on the pedestal. This releases the POWER button and removes power from the system.

2.5 COPYING AND FORMATTING

Before you begin to develop programs for the MICRAL C, you will need to format your blank diskettes and make several copies of the system diskette.

2.5.1 Formatting

All blank diskettes must be formatted prior to use. The formatting operation initializes a diskette, programming it with the proper housekeeping data in the format required for use by BAL. Following this, it is ready for use with the system.

Format your diskettes as follows:

1. With the system operating (awaiting a BAL command), insert the System diskette into Unit 0 and close the door. Make sure that the diskette is write protected.
2. Type C(cr). This brings in the formatting program and the console display is:

COMMAND :

3. For safety, remove the System diskette from Unit 0 and insert a blank diskette (or a diskette to be re-formatted) into Unit 1. Make sure that this diskette does not have a tab covering the Write Protect Notch.
4. Respond to COMMAND by typing P(cr). You must then supply data for several conditions as indicated.

COMMAND :P(cr)

OUTPUT :FLO(cr) UNIT :1(cr)

This specifies that formatting is to occur on the floppy diskette (FLO) of Unit 1. If you specified Unit 0, the System diskette would attempt to re-format itself.

5. Drive 1 will step through all the tracks as the diskette is formatted. When formatting is complete (about a minute), a new message, COMMAND : , appears on the display. If formatting cannot occur (e.g. the diskette is write protected) you will hear a steady series of beeping tones.
6. Repeat this operation from step 3 to format another diskette.
7. To terminate the operation, type M(cr) to return to BAL.

Once a diskette has been formatted and data has been recorded on it, you can erase the programs or data by copying new information on the diskette. It is not generally necessary to re-format.

2.5.2 Copying A Diskette

Prior to developing user programs, you will need to make one or more copies of your System diskette. These will be used as dedicated systems with your translated programs (see 2.7 for further information). Copy your system diskette as follows:

1. With the system operating, insert the System diskette into Drive 0.
2. Type C(cr). The copy program is loaded and the following is displayed on the screen:

COMMAND :

3. Insert a previously formatted blank diskette into Unit 1 and close the door. This diskette must not be write protected.
4. Respond to the display as follows:

COMMAND :3(cr)

5. The drives will step through all the tracks as the information on the diskette in Unit 0 is copied onto the diskette in Unit 1. After the copying is complete, about 2-3 minutes, COMMAND : again appears on the screen.

The diskette in Unit 1 is now a duplicate of the diskette in Unit 0, which is unchanged by this operation.

6. To make another copy, repeat this procedure from step 3.
7. To terminate this operation, type M(cr) to return to BAL.

Note that the COMMAND function includes other utilities in addition to Copy. They are discussed in paragraph 2.11.

2.6 CREATING A SOURCE PROGRAM

MICRAL C allows the user to create a new source program and have it syntax-checked on a line-by-line basis as it is entered. The BAL translator analyzes each instruction character by character and indicates any errors as soon as the next character has been entered.

When an error is found, the programmer can correct it immediately. See Appendix A for an explanation of the error codes. See the MICRAL C BAL Reference Manual for a complete description of the language.

Create a program as follows:

1. With the system running, insert the diskette containing your previously copied dedicated system program into Unit 0 (this becomes an Execution diskette) and a previously formatted blank diskette into Unit 1.
2. After closing the drive doors, respond to the BAL prompt by typing T(cr). BAL...T is now on the screen.
3. The translator is loaded into memory and the following message is then displayed:

NEW, OLD (N,O) :

4. To create a new program, type N(cr), then proceed to enter your BAL program from the keyboard according to the programming rules and formats defined in the MICRAL C BAL Reference Manual.

As the program is entered, it is analyzed and recorded on the diskette in Unit 1 which becomes your Source diskette.

5. When you enter each ESEG statement (end of statement), the last source code for that segment is recorded on the diskette and the following is displayed for

that segment:

PROGRAM LENGTH XX (Will be 0 if the program
cannot be translated properly.)

DATA LENGTH YY (Only at the end of Segment 0.)

6. When END is keyed as the last instruction of your program, the system writes the last of the source code on the Source diskette and returns to BAL, displaying the BAL... prompt.
7. You now have an Execution diskette in Unit 0 and a Source diskette in Unit 1. Your program is now translated and the translated binary program is recorded on the Execution diskette in Unit 0. At this point, this diskette is an Execution diskette only; the translator and utilities have been erased to provide more room for the binary file.

If there, are no errors, you can execute your translated program. If errors are detected, use the Editor to prepare a corrected Source diskette.

2.7 TRANSLATING A PROGRAM

As you no doubt noted above, the Translator is used both to create a new program and to translate an existing program. The procedure for translating an existing program is:

1. With the system operating, insert your dedicated System diskette in Unit 0; your Source diskette in Unit 1 and close the doors. Note that the dedicated System diskette may not have the Write Protect notch covered.
2. Respond to the BAL prompt by typing T(cr).
3. The Translator is loaded into memory, then several messages are displayed, requiring operator response, as follows:

Message

Response

NEW, OLD (N,O) : N This indicates a new program which is created as described in paragraph 2.6.

O An old program, already on the Source diskette. Additional information will now be requested.

ESC Return to BAL.

LIST (Y,N) : Y Yes; a program listing is output to the printer or the CRT.

N No; no listing required.

ESC Return to the New-Old question.

DEBUG (Y,N) : Y Yes; the hexadecimal address of each line of the program is printed in parentheses on the listing along with the program source code.

If a listing is not selected, the hexadecimal addresses for lines with statement numbers will be printed in a table.

N No; no hex addresses are printed.

ESC Return to the New-Old question.

OUTPUT
NUMBER

1 Output listing on CRT display.

2 Output listing on printer.

ESC Return to the New-Old question.

PART (Y,N) : Y Yes; selects the partial translation of the program -- some number of segments to be translated. (When you translate Segment 0, the entire program must be translated.)

The following additional information is required:

SEG NUMBER :N(cr) Respond to each request with the number of a segment to be translated. Respond with (cr) to begin translation.

N No; translate the complete program. When segment 0 is modified, or if variables are changed, it is suggested that the complete program be re-translated.

ESC Return to the New-Old question.

4. The source program is read from the diskette in the right hand unit, Unit 1, and translated with the resultant program recorded on the Execution diskette in Unit 0.
5. Follow each of the responses with a carriage return.

As translation occurs, syntax errors may be found. When this occurs, the system indicates the error by displaying an error number on the CRT and halting. (See the translation time error list, Appendix A.) The offending character will be enclosed in parentheses.

EXAMPLE: 25 DCL A, B2, CD
 *** (D) ERROR 61 DEBUG ADDRESS 0000

You must press ESC to continue the translation.

6. You can stop the translation at any time by pressing the Escape (ESC) key. When this occurs, program translation is stopped, and you can select one of the following options:
 - a. Continue the translation by again pressing ESC.
 - b. Change the conditions of the translation by pressing R(cr). You must then respond again to the LIST, DEBUG, and OUTPUT NUMBER questions.
 - c. Abandon the translation by first pressing R, then responding to the LIST message by pressing ESC. When NEW,OLD appears on the screen, again press ESC to return to BAL.

7. When the translation is complete, the system returns to BAL and displays:

BAL...

8. At this point you have the following:

- a. An Execution diskette containing the translated program and the run-time system program.

Note that the Execution diskette no longer contains the complete operating system. In order to provide maximum room for the translated program, all system routines, except the run-time package are erased.

- b. A Source program diskette.

9. A translated program can be executed only when it has no errors. It is not necessary to re-translate the program after execution.

2.8 PROGRAM EXECUTION

Once your program has been translated, and you have an errorless program, that program can be executed. The steps for program execution are:

1. With the system operating, insert the Execution diskette in Unit 0. If a file diskette is being used, insert it in Unit 1.
2. Close the unit's doors and type R(cr).
3. The interpreter, along with the translated object program, are loaded into memory and program execution begins.
4. During execution if the program is semantically incorrect, a run-time error will occur (see Appendix A for a list). All run-time errors (except file system instructions, if errors are controlled by the program) are fatal and cause execution to halt with the following message displayed on the screen:

ERROR N IN SEGMENT X AT ADDRESS Y

Where: N is the error number
X is the program segment number
Y is the Debug address within the segment

2.9 EDITING A PROGRAM

To edit your program, you must provide:

1. The System diskette.
2. The Source diskette.
3. A formatted diskette, either blank or a diskette to be overwritten.

The steps for program editing are:

1. With the system operating, insert the System diskette in Unit 0, close the door, and type U(cr).
2. The Utility package is loaded into memory, and the following message is displayed:

FUNCTION :

3. Remove the System diskette from Unit 0 and replace it with the blank diskette.
4. Insert the Source diskette containing the program to be edited in Unit 1.
5. Close the doors and type E(cr), to call the Editor.
6. Next respond to the following working messages, as shown:

INPUT : <u>FLO(cr)</u>	UNIT : <u>1(cr)</u>	TRACK : <u>0(cr)</u>
OUTPUT : <u>FLO(cr)</u>	UNIT : <u>0(cr)</u>	TRACK : <u>0(cr)</u>

7. The first instruction of the program is now displayed on the CRT, along with a colon prompt, as shown below:

```
PROGRAM "TEST"  
:_
```

You must respond with one of the Editor commands. They are summarized below and explained in detail in the MICRAL C BAL Reference Manual. An invalid command

will be rejected by the system. The character will not appear on the screen, but a light beeping tone will be sounded.

Command	Function
Space (press space bar)	Advance to next instruction.
M (modify)	Modify instruction presently displayed. Pressing ESC at any point moves the cursor to the end of the instruction, retaining the data passed over.
C (comment)	Similar to M, but moves cursor to the end of the instruction, so comments can be added.
R (Replace)	Delete the current instruction and replace with one or more instructions.
I (Insert)	Insert one or more instructions following the instruction being displayed.
Dn (Delete)	Delete n lines (0 ≤ n ≤ 9), beginning with the current line.
+n	Advance the display down n instructions, where n is from 1 to 9.
J(cr)	Insert source code from some other support device, defined by your response to the question INPUT : which will now be displayed. This will generally be a source file from the diskette. It must terminate with an END statement (which will not be copied).
A	Advances to the end of file. The message FILE END is displayed.
,string(cr) .string(cr) *	Used to find the occurrence of strings in text beginning from the present line. , string -- Find the first occurrence of the specified string anywhere in text. . string -- Find the first occurrence of the specified string which <u>begins</u> a line. * -- Repeat the previous string command.

Note that a string may be up to 15 characters in length.

Command	Function
S(cr)	Abort the edit, cancelling all changes to the source program. The Utility requests the next FUNCTION.
E(cr)	Write the modified source code to the new diskette. This command must be executed to properly terminate an edit and obtain the newly edited source diskette. This command can be entered at any point in the edit.

Note: Several keys are used for special functions in the Editor. They are:

Escape	-- For the M and C commands, moves the cursor to the end of a line, retaining the information passed over.
Left Arrow	-- Backs the cursor up one space and allows retyping the previous letter.
Right Arrow	-- Steps the cursor one character to the right without destroying the bypassed character.

8. When your edit is complete, type E(cr). The remainder of the original source program is copied into the new file, and the message FUNCTION appears on the display.
9. You now have an edited Source program in Unit 0 and the original Source program in Unit 1. You can then copy the edited program back to the original source diskette if you wish, as follows:

FUNCTION : <u>S(cr)</u>		
INPUT : <u>FLO(cr)</u>	UNIT : <u>0(cr)</u>	TRACK : <u>0(cr)</u>
OUTPUT : <u>FLO(cr)</u>	UNIT : <u>1(cr)</u>	TRACK : <u>0(cr)</u>
10. Copying is complete when FUNCTION is again displayed. Type M(cr) to return to BAL.
11. You now have two diskettes containing the edited Source program. This program must be translated before it can be executed.

2.10 THE DEBUG PACKAGE

Debug is a routine which allows you to execute your BAL program interactively. In normal execution of a BAL program, instructions are fetched, checked for run-time errors, and executed in sequence. Under Debug control, you can execute instructions in single step mode, insert breakpoints, examine and modify variables, etc. This is handy when you're trying to figure out why the program doesn't work the way you planned it.

Use Debug as follows:

1. With the system operating, insert your Execution diskette in Unit 0. If a file diskette is required, insert it in Unit 1. Close the doors.
2. Type D(cr). The Debug routine and your object program are loaded into memory and the following message is displayed:

DEBUG :

3. At this point, you must provide working instructions to the Debug program. A condensed table of these instructions is below. A more detailed description of these instructions can be found in the MICRAL C BAL Reference Manual.

Command	Function
D variable(cr)	Displays the present contents of the selected variable.
M var=value(cr)	Replaces the contents of the selected variable with "value".
B seg no,addr(cr)	A breakpoint is installed in program segment "seg no" at the specified address (addr). When instruction execution reaches that point in the program, control is returned to the user prior to executing the instruction. The breakpoint is removed, once it has been reached.
A seg no, addr(cr)	A permanent breakpoint is installed at the specified address in "seg no". Program execution will stop and

control is returned to the user each time this instruction is reached. This condition is reset with the C command.

- | | |
|---------------|--|
| G(cr) | Continue program execution from the current point. |
| G address(cr) | Begin program execution from the indicated point. The address corresponds to one of the addresses printed on the source listing when the Debug option is selected. |
| S(cr) | Selects single step instruction execution in which one instruction is executed each time the space bar is pressed. |
| C(cr) | Terminates single step mode and the permanent breakpoint settings. |
| Space | Pressing the space bar while in single step mode results in the execution of the next instruction in sequence. |
| E(cr) | Terminates execution of the program under Debug control and returns to BAL. |

4. When a program is running under Debug, control is returned to the operator at the first instruction, after each instruction in single step mode, each time a breakpoint is reached, and upon the detection of a run-time error. Control is also returned if you press ESC (if the program is hung up in a loop, for example).

2.11 THE UTILITY ROUTINES

There are two sets of utility routines available. They are described below.

2.11.1 Routines For Handling Source Files

These routines include the Text Editor and the Source Program Copy routine. They are used as follows:

1. With the system operating, insert your System diskette in Unit 0 and type U(cr).
2. The Utility package is loaded into memory and the system displays:

FUNCTION :

3. Type E(cr) to call the Editor. This was discussed in detail in paragraph 2.9.
4. Type S(cr) to call the Source Program Copy routine. Then respond to the display messages as shown below.

a. To copy from one floppy disk to another

```

INPUT :FLO(cr)    UNIT:0 or 1(cr)    TRACK :0(cr)
OUTPUT :FLO(cr)    UNIT :0 or 1(cr)    TRACK :0(cr)

```

Naturally, you must have the proper diskettes in the drives for this operation. You can copy from a source diskette in Unit 0 to a formatted diskette in Unit 1 or vice versa.

You will normally copy the system program from track 0 on one unit to track 0 on the other unit because that's where the Translator expects to find it. You can copy to other tracks, but be careful that you don't overlay valid data.

b. To copy from floppy to CRT or printer

```

INPUT :FLO(cr)    UNIT:0 or 1(cr)    TRACK :Y(cr)
OUTPUT :STY(cr)    (To the CRT)
      or
OUTPUT :IMP(cr)    OUTPUT NUMBER :Z(cr)

```

Where: Y = Track number - normally track 0 for the system program.

Z = 1 - Output to CRT

2 - Output to printer

5. Type M(cr) to return to BAL.

2.11.2. Routines For Handling Various Types of Files

These utilities include routines for copying files, dumping data and formatting.

1. With the system operating, insert a System diskette into Unit 0, close the door, and type C(cr).
2. The utility will be loaded into memory, and the following is displayed:

COMMAND :

3. Respond to the display with one of the following valid codes after loading the required diskettes in the drives:

Code	Function
3(cr)	Copies the complete contents of the diskette in Unit 0 onto the diskette in Unit 1.
R(cr)	Copies the complete contents of one diskette onto another. You must supply system information as follows: <div style="margin-left: 40px;"> INPUT :<u>FLO(cr)</u> UNIT :<u>0 or 1(cr)</u> OUTPUT :<u>FLO(cr)</u> UNIT :<u>0 or 1(cr)</u> </div>
P(cr)	Formats a diskette. In response to command P, the system requires the following information: <div style="margin-left: 40px;"> OUTPUT :<u>FLO(cr)</u> UNIT :<u>X(cr)</u> </div> <p>You can insert a diskette in either drive and format it using this command.</p>
D(cr)	Dumps the selected sectors from a floppy to the CRT or printer. To use this command, insert a diskette to be dumped into one drive and type D(cr). Then supply the necessary system information as shown below.

INPUT :FLO(cr) UNIT :X(cr) FIRST
 SECTOR :AA(cr) NB SECTORS :BB(cr)
 OUTPUT :STY(cr) (Dump to CRT)

or

OUTPUT :IMP(cr) OUTPUT NUMBER :Z(cr)

Where: X = Unit 1 or 0
 AA = Starting sector number
 BB = Number of sectors (decimal or hex, preceded by /)
 Z = 1 - CRT; 2 - printer

This command dumps the required number of sectors of data (BB) from Unit X, starting with sector AA onto the designated device.

Data is dumped 16 characters per line, with the hexadecimal numbers on the left and their ASCII equivalents on the right. Unprintable ASCII characters are represented as a dot, ".".

Data is output sector by sector. If output is to the CRT, the system halts after each sector is output. Press any key (except ESC) to continue with the next sector. If you press ESC, the dump is aborted and you return to the COMMAND message.

L(cr) Free Format -- This command copies the specified files, no matter what format they are in, onto the specified output device.

The procedure for this command is identical with D, above.

M(cr) Return to BAL.

4. At any point, you can abandon your selection and return to COMMAND by pressing the Escape Key (ESC). The system will print ABORT, then return to COMMAND.

CHAPTER 3. OPERATING INSTRUCTIONS -- FORTRAN

3.1 INTRODUCTION

This chapter contains a complete description of operating instructions for FORT//80*, the FORTRAN IV Compiler available for the MICRAL C.

For a description of the FORTRAN language available in FORT//80, refer to the FORT//80 FORTRAN IV Language Manual.

3.2 LOADING THE SYSTEM

1. Power the MICRAL C system up by pressing the green POWER button (located on the upper left hand corner of the pedestal) until it locks into place.
2. The red and green indicators on the control panel will light, and the CRT will come on, provided that the CRT circuit breaker (located on the left rear of the unit) is on. Use the rotary CRT intensity switch to adjust the brightness of the display.
3. Insert the FORTRAN System diskette in the left diskette drive (Unit 0) and lift the DEPART switch on the system control panel.
4. The system will load and a dollar sign (\$) prompt will be displayed. Respond to this prompt as follows to load the resident program and the FORTRAN compiler:

\$B:0,4(cr)

\$B:0,15(cr)

5. When these programs have been loaded, FORT: is displayed. You may now choose one of three options:

EDIT -- Used to create and/or change your FORTRAN source program. This is the same editor used with BAL.
COMP -- Compile - This selects the compilation of your FORTRAN source program.
EXEC -- Execute - This function selects the execution of a previously compiled FORTRAN program.

*Distributed under license from Unified Technologies, Inc. of Canada.

3.3 EDITING A FORTRAN PROGRAM

1. REMOVE your FORTRAN System diskette from Unit 0 and place it in Unit 1. Place your Source diskette in Unit 0.
2. You must then respond to the FORT prompt and supply a job number as shown below:

```

FORT:EDIT(cr)
FUNCTION:7(cr)
```

3. The Editor is then loaded and the first line of your source program appears on the screen. You then create or edit your program using the MICRAL Editor commands as described in paragraph 2.9.

There are two areas of difference, which are:

- a. For FORTRAN versions 3.2E and above, a tab function is available. To tab to column 7, press the ETX key after keying the M, I, or R commands, but before inserting data.
- b. Enter your FORTRAN program per standard FORTRAN conventions. The program must terminate with an END statement. The Editor then requires an EOF (end-of-file) to be entered on the next line, as shown below. (This is necessary only when originally creating the program.)

```

.
.
100  CONTINUE(cr)
      END 10H(cr)
      EOF(cr)
```

The EOF is for Editor use only. It is not copied as a part of your FORTRAN source file.

4. When you finish editing your program, terminate the edit with an E(cr), as normal. Your Source diskette then contains the newly edited version of your program. The display will be: FUNCTION :
5. Return to the FORT prompt by typing M(cr).

3.4 PROGRAM COMPILATION

1. Make sure that you have a FORTRAN System diskette in Unit 1 and a FORTRAN Source program diskette in Unit 0.
2. You must then respond to the FORT prompt as shown:

FORT:COMP(cr)

3. The system now requests a job number. This allows you to select some combination of source program listing and binary object file, as shown in table 3-1.

<u>Table 3-1. FORTRAN Compilation Job Numbers</u>	
<u>Job Number</u>	<u>Option Selected</u>
1	Source program listing, interleaved with the binary code generated for each line.
2	Binary object file only
3	Interleaved listing and binary object file.
4	Source listing only
5	Source listing and binary object file

The source listing is output on the system printer. The binary object file is recorded on the same diskette as the source file.

4. Including the job number, respond to system messages as follows:

JOB NUMBER:X(cr)
SOURCE UNIT:0(cr) TRACK:0(cr)
PUNCH UNIT:0(cr) TRACK:/18(cr)

5. The message "UTI FORT//80 COMPILER VX.Y* is now displayed, and compilation occurs. FORT//80 is a two pass compiler. After the first pass, PHASE 2 is displayed on the CRT, and the source listing is printed (if requested).

When a FORTRAN error is found, an error indication is printed in the source listing immediately following the line in error. This includes an error number and an indication of the character(s) in the statement near which the error was found. Refer to the FORT//80 Language Manual for a complete list of FORTRAN error codes.

* X.Y is the version number, i.e., 3.2B.

6. When compilation is complete, the total number of errors is printed, and the system will again request JOB NUMBER. Type M(cr) to return to the FORT prompt.

3.5 PROGRAM EXECUTION

1. Place your Source diskette containing the compiled object program in Unit 0. Note that you do not need to have your FORTRAN System diskette in the system, as FORT//80 produces directly executable object files.

2. Respond to the FORT prompt as follows:

FORT:EXEC(cr)

3. The object program is then loaded into memory. Execution begins automatically if a transfer address follows the END instruction of the source program.

Example: END 10H (The program begins at loc. 10 Hex.)

4. If no transfer address was specified, start the program as follows:

G:XXXX(cr)

Where XXXX is the starting address of the program as specified in the COMPILER(1) directive. If this directive is omitted, the program starting address defaults to 0000.

5. The program can be halted by pressing the INIT key on the system control panel. When this occurs, or when a STOP instruction is executed, the FORT prompt is again displayed and the system is ready for another command.

APPENDIX A. MICRAL BAL ERROR LIST

A.1 GENERAL

When the translator detects an error, either during translation or the execution of a program, a message is displayed on the CRT screen in the following format:

ERROR N IN SEGMENT XX AT ADDRESS YYYY

Where: N --- The error code as listed below.
XX -- The program segment which was being executed or translated.
YYYY -- The debug address of the instruction which is in error.

Note: The list of errors below is inclusive for all versions of BAL. Certain of these error codes are not applicable to MICRAL C, and will never appear on your display.

A.2 ERRORS FOUND DURING EXECUTION OF A PROGRAM

<u>Error Code</u>	<u>Explanation</u>
10	Specified segment not in the user program
11	Data field is too small -- attempt to read non-existent data
12	Index is zero
13	Index is too large for the declared table size
14	Index is non-binary (table or string)
15	RETURN point in a program is incorrect
16	Overflow of a variable
17	The stack of the GOSUB or LDGO.SEG has overflowed
18	Arithmetic overflow
19	Undefined variable
20	Variable is not in floating point format (BCD)
21	Load segment (LDGO.SEG X) is incorrect
22	BUFIN or BUFOUT error -- BUFIN & BUFOUT applicable only to disk and floppy
23	Incorrect peripheral number specified (ASK or PRINT)
24	Format error
25	Common memory overflow
26	Lock number too large
27	Memory variable too large

<u>Error Code</u>	<u>Explanation</u>
28	Individual memory overflow
29	Common variable declaration incorrect
30	Common variable table overflow
31	Undefined operation code
32	Attempt to write a protected common variable

A.3 ERRORS FOUND DURING TRANSLATION OF A PROGRAM

<u>Error Code</u>	<u>Explanation</u>
50	Keyword incorrect (READ, PRINT, GOTO, etc.)
51	Error in the form of the instruction (Incorrect label)
52	Value specified is not binary
53	Number of segments is not correct
54	The end of the instruction is incorrect
55	FOR/NEXT loop is incorrect
56	Syntax error
57	Operator is incorrect (+, -, *, etc)
58	Type of variable is incorrect (string, BCD, binary)
59	Format error
60	Support variable incorrect (FIELD, EQUIVALENCE)
61	DCL (declaration) error
62	The binary code generated for this in- struction is too large; the instruction must be shortened
63	String is incorrect
64	BCD is incorrect
65	Overflow of stack for the nested FOR statement; number of nested loops must be reduced
66	Overflow of the individual memory; too many variable declarations
67	The area selected for writing is already in use for a BAL program

A.4 ERRORS FOUND WHEN USING PERIPHERALS

<u>Error Code</u>	<u>Explanation</u>
01	The disc is not ready
02	Read or write error
03	Track positioning error
04	Parameter call error

A.4 ERRORS FOUND WHEN USING PERIPHERALS, CONT'D.

<u>Error Code</u>	<u>Explanation</u>
05	Peripheral is write protected
06	Disc hardware error
08	File is not in binary format
40	Specified peripheral not in the system
41	Location of the volume unknown
44	Volume does not belong to the file system

Note: Certain errors do not result in immediate abortion of the operation. In these cases, the system retries the operation and sounds a beeping tone to alert the operator of the retry. You may abandon these retry attempts by pressing the ESC key. This procedure is applicable to errors 1,2,3,5,6 and 25.

APPENDIX B. MICRAL FILE SYSTEM RESPONSES

B.1 TABLE OF RESPONSES

<u>Hex</u>	<u>Decimal</u>	<u>Explanation</u>
/0	0	Request correctly executed
/1	1	Peripheral not ready
/2	2	Read or write error
/3	3	Track positioning error
/4	4	Incorrect parameter call
/5	5	Peripheral is write protected
/6	6	Peripheral hardware error
/7	7	Start of file detected (tape)
/8	8	End of file detected (tape)
/9	9	Peripheral busy
/40	64	Peripheral or file system resource non-existent
/41	65	Local volume undefined
/42	66	Function undefined
/43	67	Volume unchangeable because it still contains files
/44	68	Volume does not belong to file system
/45	69	Volume not mounted
/46	70	Name of the file does not exist on the volume
/47	71	Incorrect file opening key
/48	72	File open
/49	73	Incorrect logical number
/50	80	Logical number not assigned
/51	81	Relative file not open for writing
/52	82	File name already exists on this volume
/53	83	The volume cannot contain a new file
/54	84	Unknown disc address
/55	85	Volume overflow
/56	86	Incoherent file
/57	87	Logical number already assigned
/58	88	The file system cannot accept the open- ing of a new file
/59	89	The volume is used, therefore not re- placeable by another
/60	96	Incorrect file type
/61	97	File not open for writing
/62	98	Sequential file not open for writing
/63	99	Beginning of file
/64	100	End of file

B.1 FILE SYSTEM TABLE OF RESPONSES, CONT'D.

<u>Hex</u>	<u>Decimal</u>	<u>Explanation</u>
/65	101	Loss of information on reading; your input buffer contains only the beginning of the recorded data
/66	102	Length of the key is incorrect
/67	103	Article found, but does not appear in any of the indexes requested
/68	104	Article not found
/69	105	No index
/70	112	Article already exists
/71	113	Article in course of modification
/72	114	Article not authorized for modification
/73	115	Writing or destruction of a file is not permitted because that file is not the last on the tape
/74	116	Attempt to open a file while it is in safeguard

USER RESPONSE FORM

TM 1001 -- MICRAL C OPERATOR'S MANUAL

Please use this form to request future updates and/or record your comments, suggestions, etc. concerning this manual.

Date _____

Name _____

Address _____

City _____ State _____ Zip _____

☐ Check here if you wish your name placed on the mailing list for future updates to this document.

RETURN TO:
R2E OF AMERICA
3406 University Ave., S.E.
Minneapolis, Minnesota 55414